CURRENT LISTING OF CLAIMS

The listing of claims below replaces all prior versions, and listings, of claims:

1. (Previously Presented) A method for use in a database system having plural nodes, comprising:

storing a materialized join view based on at least two base relations;

storing at least one auxiliary relation containing one or more attributes of one of the base relations, the auxiliary relation partitioned across the plural nodes according to a join attribute; and

updating the at least one auxiliary relation in response to modification of the one base relation.

2. (Previously Presented) The method of claim 1, further comprising storing the one base relation that is partitioned across the plural nodes according to an attribute that is different from the join attribute.

3. (Original) The method of claim 2, further comprising:

receiving a tuple into the database system;

storing the tuple in one of the at least two base relations;

storing the tuple in the auxiliary relation; and

using the auxiliary relation to determine whether to update the materialized join view.

4. (Previously Presented) A method comprising:

receiving a first tuple into a base relation at a first node of a parallel database system having plural nodes, wherein the first tuple comprises a join attribute and the base relation is partitioned across the nodes according to an attribute different from the join attribute;

storing the first tuple in an auxiliary relation at a second node of the parallel database system, wherein the auxiliary relation is partitioned across the nodes of the database system according to the join attribute;

identifying second tuples of a second relation;

joining the first tuple with the second tuples based on the join attribute to produce join results; and

storing the join results in a join view.

5. (Cancelled)

6. (Cancelled)

7. (Previously Presented) The method of claim 4, wherein storing the first tuple in an auxiliary relation at a second node comprises:

determining that a join view definition excludes an attribute of the first tuple; and

not storing the excluded attribute in the auxiliary relation.

8. (Previously Presented) The method of claim 4, further comprising:

receiving a third tuple into the base relation;

determining that a join view definition includes a condition on one of the attributes of the third tuple; and

determining that the condition is not met by the one of the attributes of the third tuple; and

not storing the second tuple in the auxiliary relation.

9. - 12. (Cancelled)

13. (Previously Presented) A database system comprising:

storage modules to store base relations and at least a first auxiliary relation corresponding to a first one of the base relations, the first auxiliary relation containing one or more attributes of the first base relation, the first auxiliary relation partitioned across the storage modules differently than the first base relation, the storage modules further to store a join view based on a join of the base relations; and

a controller adapted to update the join view using at least the first auxiliary relation.

14. (Previously Presented) The database system of claim 13, wherein the controller is further adapted to receive a tuple and store the tuple in the first base relation and in the first auxiliary relation.

15. (Original) The database system of claim 14, wherein the controller is further adapted to not update the join view after receiving some tuples.

16. (Previously Presented) An article comprising a medium storing instructions for enabling a processor-based system having plural nodes to:

store a join view to store join results of a join of at least first and second base relations based on a join condition including a first attribute of the first base relation and a second attribute of the second base relation;

receive a first tuple into the first base relation at a first node, wherein the first tuple comprises the first attribute and the first base relation is partitioned across the plural nodes according to an attribute other than the first attribute;

store the first tuple in a first auxiliary relation at a second node, wherein the first auxiliary relation is partitioned across the plural nodes according to the first attribute;

identify second tuples of the second base relation; and

join the first tuple with the second tuples to produce join results for updating the join view.

17. (Previously Presented) The article of claim 16, further storing instructions for enabling the processor-based system to:

compare the second attributes of the second tuples with the first attribute of the first tuple to produce the join results for updating the join view.

18. (Previously Presented) The article of claim 16, further storing instructions for enabling the processor-based system to:

determine that a join view definition excludes an attribute of the first tuple; and

not store the excluded attribute in the first auxiliary relation.

19. (Previously Presented) The article of claim 16, further storing instructions for enabling the processor-based system to:

determine that a join view definition includes a condition on one of the attributes of the first base relation; and

identify the one attribute in a received third tuple;

determine that the condition is not met by the received third tuple; and

not store the received third tuple in the first auxiliary relation.

20. - 23. (Cancelled)

24. (Previously Presented) The method of claim 1, wherein storing the materialized join view comprises storing a materialized join view containing tuples derived from a join of the at least two base relations.

25. (Previously Presented) The method of claim 24, wherein the one of the base relations comprises a first base relation, the method further comprising:

partitioning the first base relation across the plural nodes in a first way; and

partitioning the auxiliary relation across the plural nodes in a second, different way according to the join attribute.

26. (Previously Presented) The method of claim 25, further comprising:

receiving a first tuple to insert into the one of the base relations;

inserting the first tuple into the first base relation; and

inserting at least a portion of the first tuple into the auxiliary relation.

27. (Previously Presented) The method of claim 26, further comprising:

distributing the first tuple from a first one of the plural nodes to a second one of the plural nodes, wherein inserting the first tuple into the auxiliary relation is performed at the second node; and

joining the first tuple with at least a second tuple associated with a second one of the base relations in the second node.

28. (Previously Presented) The method of claim 27, wherein inserting the first tuple into the first base relation is performed at the first node, wherein joining the first tuple with at least the second tuple is performed at the second node instead of the first node.

29. (Previously Presented) The method of claim 28, further comprising storing a second auxiliary relation containing one or more attributes of the second one of the base relations, the second auxiliary relation partitioned according to a join attribute.

30. (Previously Presented) The method of claim 29, wherein joining the first tuple with the second tuple comprises joining the first tuple with the second tuple contained in the second auxiliary relation.

31. (Previously Presented) The method of claim 30, further comprising using a result of the join of the first tuple with the second tuple in the second auxiliary relation to update the materialized join view.

32. (Previously Presented) The method of claim 1, further comprising maintaining an index on the auxiliary relation.

33. (Previously Presented) The method of claim 4, wherein joining the first tuple with the second tuples is performed at the second node.

34. (Previously Presented) The database system of claim 13, further comprising plural nodes, the storage modules in respective plural nodes, and the controller comprises plural processors in respective nodes.

35. (Previously Presented) The database system of claim 34, wherein a first one of the nodes is adapted to receive a first tuple,

the processor in the first node to insert the first tuple into the first base relation, and

the processor in a second one of the nodes to insert the first tuple into the first auxiliary relation.

36. (Previously Presented) The database system of claim 35, wherein the first node is adapted to distribute the first tuple to the second node.

37. (Previously Presented) The database system of claim 36, the storage modules in the second node to store a portion of a second auxiliary relation to store one or more attributes of a second one of the base relations, the storage module in the second node to store a portion of the join view that is distributed across the plural storage modules, and the processor in the second node to join the first tuple with tuples in the portion of the second auxiliary relation to update the portion of the join view in the storage module of the second node.

38. (Previously Presented) The database system of claim 37, the join view to store join results based on a join condition including at least a first attribute of the first base relation and a second attribute of the second base relation, wherein the first auxiliary relation is partitioned across the storage modules according to the first attribute, and the second auxiliary relation is partitioned across the storage modules according to the second attribute.

39. (Previously Presented) The database system of claim 38, the storage modules to store a first index on the first auxiliary relation, and the storage modules to store a second index on the second auxiliary relation.

40. (Previously Presented) The database system of claim 13, the join view to store results of a join of the base relations based on a join condition and selection condition, the controller to store tuples of the first base relation that satisfy the selection condition in the first auxiliary relation, and the controller to not store tuples of the first base relation that do not satisfy the selection condition in the first auxiliary relation.

41. (Previously Presented) The database system of claim 13, the join view to store results of a join of the base relations based on a query containing a select clause and a join condition, the select clause specifying one or more attributes of the first base relation,

the controller to store the one or more attributes of the first base relation specified by the select clause in the first auxiliary relation, and the controller to not store other attributes of the first base relation not specified by the select clause in the first auxiliary relation.

42. (Previously Presented) The database system of claim 13, the join view to store results of a join of at least the first base relation and a second base relation based on a join condition including a first attribute of the first base relation and a second attribute of the second base relation, wherein the first attribute is a key of the first base relation and the second attribute is a foreign key of the second base relation that references the first attribute,

the controller to, in response to detecting that the first attribute is a key of the first base relation and that the second attribute is a foreign key of the second base relation that references the first attribute, create the first auxiliary relation to store the one or more tuples of the first base relation but to not create a second auxiliary relation to store tuples of the second base relation.

43. (Previously Presented) The article of claim 16, wherein the processor-based system comprises plural storage modules, the first base relation partitioned across the storage modules according to the attribute other than the first attribute, and

the first auxiliary relation partitioned across the storage modules according to the first attribute.

44. (Previously Presented) The article of claim 43, wherein the plural nodes contain the storage modules and plural processors, wherein the instructions when executed cause the processor-based system to further distribute the first tuple from the first node to the second node.

45. (Previously Presented) The article of claim 44, wherein the instructions when executed cause the processor-based system to further:

store a second auxiliary relation containing one or more tuples of the second base relation;

partition the second auxiliary relation across the storage modules according to the second attribute; and

partition the second base relation across the storage modules according to an attribute of the second base relation other than the second attribute.

46. (Previously Presented) The article of claim 45, wherein identifying the tuples of the second base relation comprises identifying the tuples of the second auxiliary base relation, and wherein joining the first tuple with the second tuples comprises joining the first tuple with the second tuples of the second auxiliary relation.

47. (Previously Presented) The article of claim 16, wherein the join view stores join results of the join of the at least first and second base relations based on a query containing the join condition, wherein the instructions when executed cause the processor-based system to further determine whether the query specifies one or more elements that enable storage of less than the entire first base relation in the first auxiliary relation.

48. (Previously Presented) The article of claim 47, wherein the first auxiliary relation stores the entire first base relation in response to determining the one or more elements do not exist.

49. (Previously Presented) The article of claim 48, wherein the one or more elements comprise a selection condition in a WHERE clause of the query.

50. (Previously Presented) The article of claim 48, wherein the one or more elements comprise less than all of the attributes of the first base relation specified by a select clause in the query.